编程题

**过程/函数**

1．创建一个以员工号为参数，返回该员工所在部门平均工资的函数。编写PL/SQL块，输出7788号员工所在部门的平均工资。

Create or Replace function func\_retavgsal(

p\_empno emp.empno%type)

return emp.sal%type

as

v\_deptno emp.deptno%type;

v\_avgsal emp.sal%type;

begin

select deptno into v\_deptno from emp where empno=p\_empno;

select avg(sal) into v\_avgsal from emp where deptno=v\_deptno;

return v\_avgsal;

exception

when no\_data\_found then

dbms\_output.put\_line('there is not such an employee');

end func\_retavgsal;

/

**调用:**

Begin

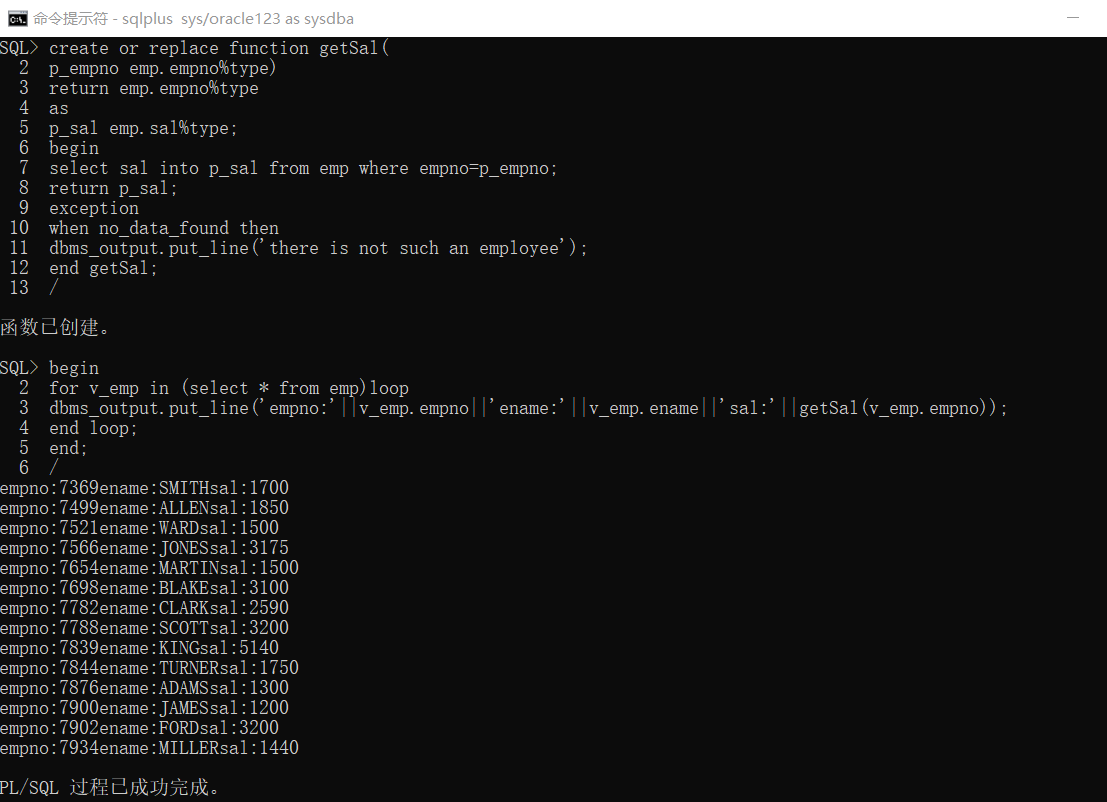
dbms\_output.put\_line(func\_retavgsal(7788));

End;

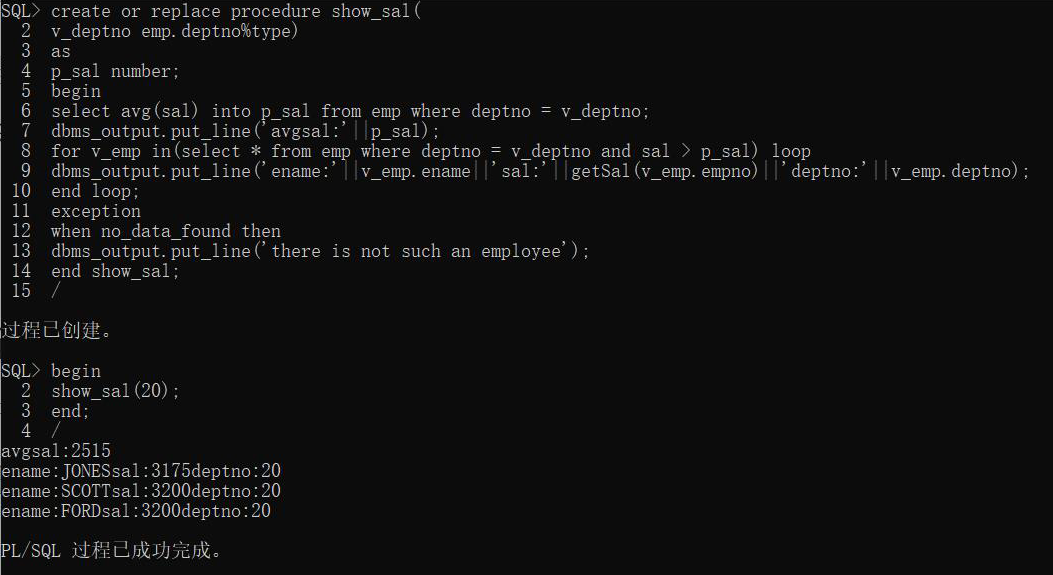
/

2．创建一个函数，以员工号为参数，返回该员工的工资。并编写PL/SQL块，

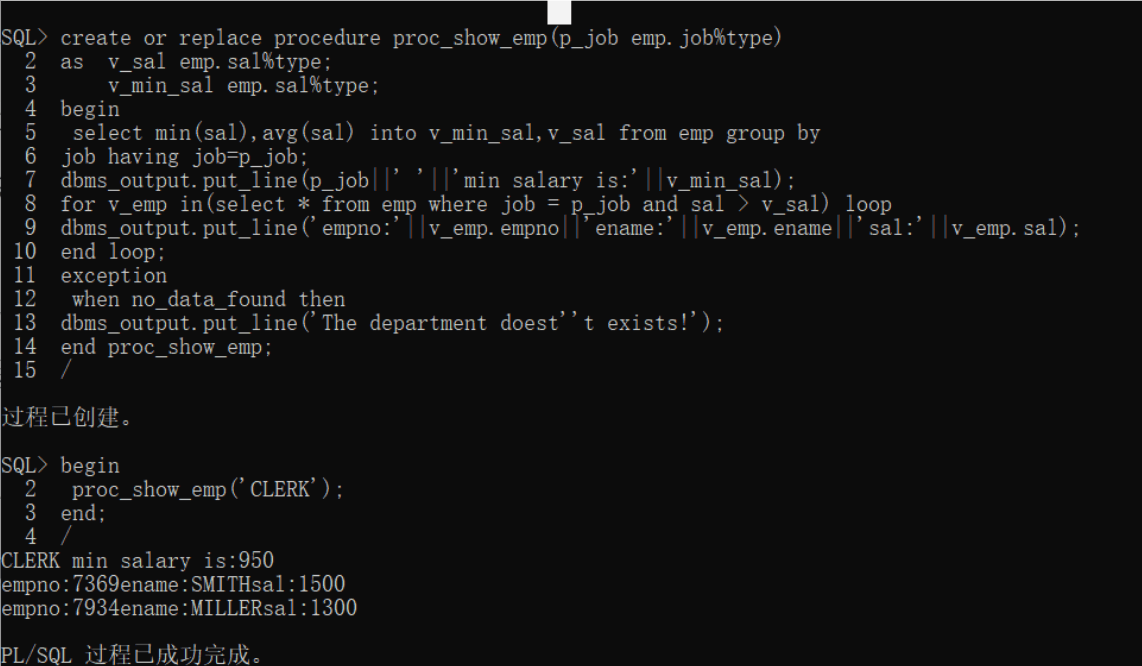
调用该函数求出所有员工的员工号，员工姓名和工资。



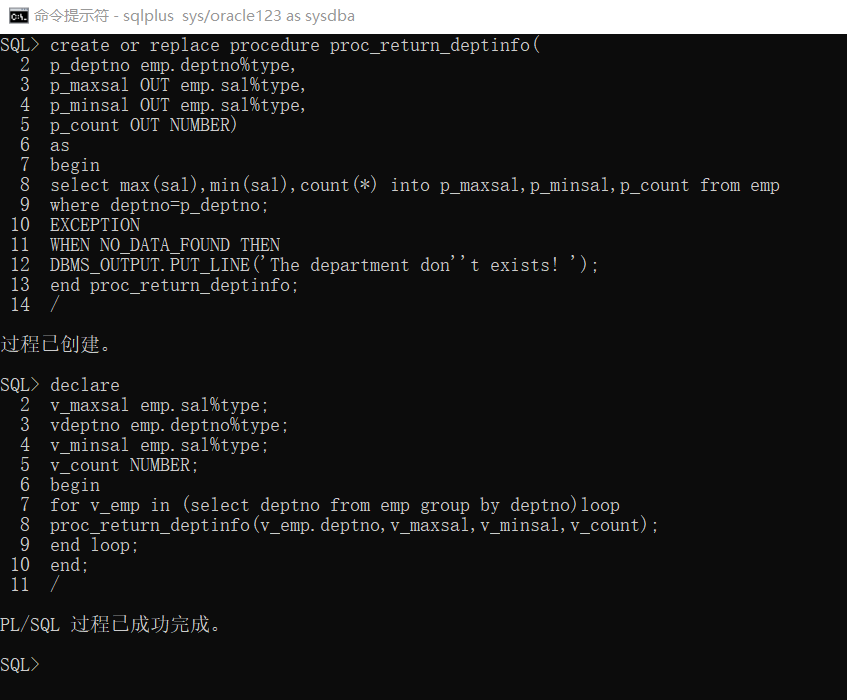
3．创建一个存储过程，以部门号为参数，查询该部门的平均工资，并输出该部门中比平均工资高的员工姓名、工资及所在部门编号。编写PL/SQL块，调用此过程求出20号部门平均工资以及20号部门中比平均工资高的员工号、员工名和工资。



4．创建一个存储过程，以工种（job）为参数，查询该工种的最低工资，并输出该工种中比平均工资高的员工号、员工名，工资。编写PL/SQL块，调用此过程求出‘CLERK’工种中比平均工资高的员工号、员工名和工资。



5．创建一个存储过程，以部门号为参数，返回该部门的人数和最高工资和最低工资。编写PL/SQL块求出所有部门的人数、最高工资和最低工资。



6．创建一个存储过程，以员工号为参数，修改该员工的工资。若该员工属于10号部门，则工资增加140元；若属于20号部门，则工资增加200元；若属于30号部门，则工资增加250元；若属于其他部门，则工资增长300元。调用该过程为所有员工修改工资。

Create or Replace procedure pro\_updatesal(

p\_empno emp.empno%type)

as

v\_deptno emp.deptno%type;

v\_inc number;

begin

select deptno into v\_deptno from emp where empno=p\_empno;

case v\_deptno

when 10 then v\_inc:=140;

when 20 then v\_inc:=200;

when 30 then v\_inc:=250;

else v\_inc:=300;

end case;

update emp set sal=sal+v\_inc where empno=p\_empno;

exception

when no\_data\_found then

dbms\_output.put\_line('thereas is not such an employees');

end;

/

**调用：**

Begin

For v\_emp in(select \* from emp) LOOP

pro\_updatesal(v\_emp.empno);

End LOOP;

End;

/

**触发器**

1．创建一个触发器，禁止在休息日改变雇员信息。

Create or Replace trigger trg\_emp1

before insert or update or delete on employees

declare

begin

if to\_char(sysdate,'DY’,’NLS\_DATE\_LANGUAGE=AMERICAN’) IN

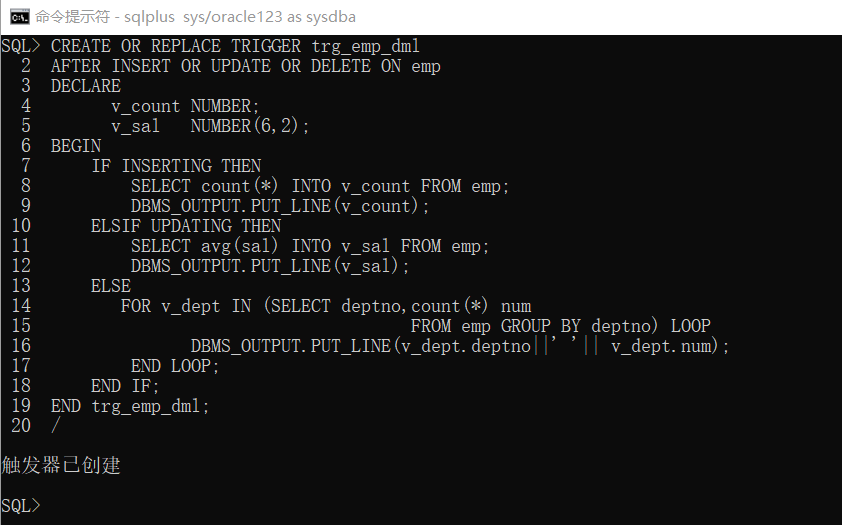
(‘SAT’,’SUN’) then

raise\_application\_error(-20005,’休息日，不允许修改EMP表');

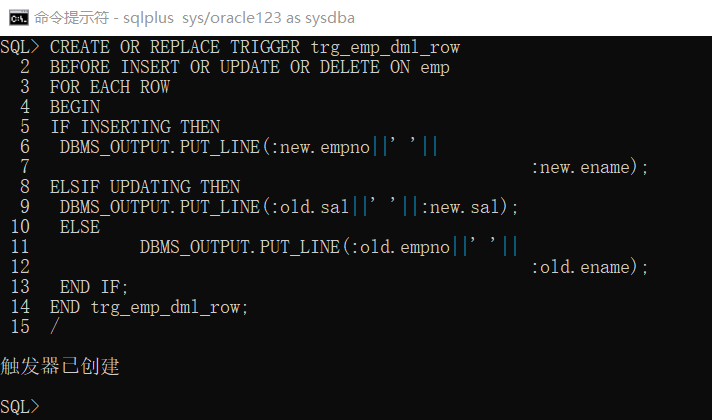
end if;

end;

2．为emp表创建一个触发器，当执行插入操作时，统计操作后员工人数；当执行更新工资操作时，统计更新后员工平均工资；当执行删除操作时，统计删除后各个部门的人数。



3．为employees表创建一个触发器，当插入新员工时显示新员工的员工号、员工名；当更新员工工资时，显示修改前后员工工资；当删除员工时，显示被删除的员工号、员工名。



4．创建一个insert触发器，当在员工表EMP中插入一条新记录时，不允许在员工编号中出现重复的编号或出现空值。

create or replace trigger trg\_emp2

before insert on emp

for each row

DECLARE

v\_count number;

begin

select count(\*) into v\_count from emp where EMPNO=:new.EMPNO ;

if v\_count =1 then

RAISE\_APPLICATION\_ERROR(-20001,'职工编号存在，不能插入数据');

ELSIF :new.empno is NULL then

RAISE\_APPLICATION\_ERROR(-20002,'职工号不能为空！');

end if;

end trg\_emp2 ;

5．创建一个insert触发器，当在员工表EMP中插入一条新记录时，Deptno(部门号码)必须是已经存在的部门号码，且Sal应该2000~8000之间。

create or replace trigger trg\_emp3

before insert on emp

for each row

DECLARE

v\_count number;

begin

select count(\*) into v\_count from dept where deptno=:new.deptno ;

if v\_count =0 then

RAISE\_APPLICATION\_ERROR(-20001,’部门号不存在，不能插入数据');

Elsif :new.sal<2000 or :new.sal>8000 then

RAISE\_APPLICATION\_ERROR(-20002,’工资必须在2000-8000之间！ ');

end if;

end trg\_emp3 ;

6．创建一个insert触发器，当在员工表EMP中插入一条新记录时，给出‘你已经插入了一条新记录！！！’的提示信息。

create or replace trigger trg\_insert

before insert on emp

begin

RAISE\_APPLICATION\_ERROR(-20002, '你已经插入记录了！');

end;

7．创建一个触发器，禁止用户删除DEPT表中的记录。

create or replace trigger trg\_delete

before delete on dept

begin

RAISE\_APPLICATION\_ERROR(-20002, '不允许删除dept表的记录！');

end;